



*Substitute Specification - Revised Copy
U.S. Application No. 10/713,017*

ARTIFICIAL INTELLIGENCE SYSTEM AND CORRESPONDING
METHODS FOR SELECTING MACHINABILITY DATA

FIELD OF INVENTION

[0001] The invention relates to an artificial intelligence system and corresponding methods for selecting optimal machining parameters especially for cutting speed and feed rate control, and more particularly applicable to computer controlled milling machines, drilling machines, grinding machines, turning machining and other such machines.

BACKGROUND OF INVENTION

[0002] Machinability data is never defined precisely in a scientific way. Machinability data consists of selection of appropriate cutting tools, as well as machining parameters including cutting speed, feed rate and depth of cut. It plays an important role in the efficient utilization of machine tools and thus significantly influences the overall manufacturing costs. It has become more and more important especially in computer aided design and computer aided manufacturing.

[0003] The main objective of machining is to satisfy the demand of the work piece dimensions and its surface finish. The machinist is subjected to a wide range of possible speed, feed rate, depth of cut, cutting tool selection, work piece material, and other machining parameters. The application of machining economics to shop floor practice has been limited and the correct choice of machining conditions is normally overlooked.

[0004] The general practice of experienced machinists is captured, formalized and made available in the form of machining data handbooks, data sheets and charts, nomograms and slide rules. Metcut's Machining Data Handbook (MDH) is among the common sources. The data in

the handbook were separated into different types of machining processes, such as face milling, end milling, turning, boring, etc. The data were grouped according to the types of work piece material and the respective hardness. The handbook provides the speed of cut and the feed rate corresponding to the depth of cut. For each work piece material and hardness, one may scan through different possible tool materials. Then, speed and feed rate are selected according to a tool and work-piece material combination, depth of cut, and finishing condition.

[0005] Although the handbook approach is often a logical and effective solution to the requirements of machinability data, it has the following limitations: 1) handbook recommendations represent a starting set of cutting conditions and hence tend to be conservative in order to cope with a worst-case machining scenario; 2) handbook data only applies to a particular machining situation. This data may not be suitable for slightly different machining situation; and 3) handbooks are manually input-output oriented, and hence lack compatibility with the objective of integrated automation of the manufacturing system.

[0006] Artificial intelligence (AI) is normally referred as any operation being carried out based on the wistful hope that at least some of the flexibility and power of the human brain can be reproduced by artificial means. The characteristics of brain function have inspired the development of artificial neural network. The way of human decision making and analysis in real life has inspired the development of fuzzy logic. The experience of machining operators may be expressed in terms of fuzzy logic and/or modeled with artificial neural networks.

[0007] A turning operation of wrought carbon steel with different cutting tools has been modeled successfully with artificial neural networks and fuzzy reasoning, and promising results have been shown in prior-arts by the inventor. Metcut's Machining Data Handbook was chosen as the reference for validation and data source training. The methods are further generalized in the present invention to cover all machining process with different work piece materials and cutting tools.

SUMMARY OF INVENTION

[0008] Accordingly, it is a first object of the present invention to provide an artificial intelligence system and corresponding artificial intelligence methods which provides a better solution compared to traditional methods for selecting optimal machinability data.

[0009] The present invention is described herein according to three main embodiments. The proposed systems works with different, or a combination of different, AI methods. Among the AI methods are Fuzzy Logic, Neural Networks and Genetic Algorithms. The embodiment will carry different function for different AI models.

[0010] In a first aspect of the present invention, the system comprises an Input Component, an Inference Component, and an Output Component. The Input Component comprises different input interfaces, and means to store and manipulate collected inputs. The corresponding input manipulation depends on the method incorporated in the inference engine. The Inference Component employs an AI inference engine that will process the input and prescribe the outputs in respective form depending on the AI model incorporated. The yielded outputs, which are recommended initial cutting parameters, will be further processed by the Output Component. The output will then be further processed and converted into a required form and format to be fed into the machine for further action by the machine.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] FIG. 1 is a block diagram of an artificial intelligence system for selecting machinability data.

[0012] FIG. 2 is a table describing an inputs fuzzy expression

[0013] FIG. 3 is a table describing an output fuzzy expression for a cutting speed fuzzy model.

[0014] FIG. 4 is a table describing min-max ranges of fuzzy membership functions for a cutting speed fuzzy model.

[0015] FIG. 5 is a table describing fuzzy rules for a high-speed steel fuzzy model.

[0016] FIG. 6 is a table describing fuzzy rules for an uncoated brazed carbide fuzzy model.

[0017] FIG. 7 is a table describing fuzzy rules for an uncoated indexable carbide fuzzy model.

[0018] FIG. 8 is a table describing fuzzy rules for a coated carbide fuzzy model.

[0019] FIG. 9 is a table describing input and output membership functions for a high-speed steel feed rate fuzzy model.

[0020] FIG. 10 is a flow chart for a general system of genetic optimization of fuzzy rules.

[0021] FIG. 11 is a block diagram of a simple non-linear neural network of machinability data for turning .

DETAILED DESCRIPTION OF THE INVENTION

[0022] With reference to FIG. 1, the present invention incorporates is the use of Artificial Intelligence techniques for selecting optimal machining parameters in conventional machining operations. The device comprising three main components which are the input component, the inference component and the output component.

[0023] The inputs to the input component are the machining operations, work piece material, machining tool type and depth of cut. The outputs of the output component are the machining parameters comprising the optimal cutting speed and feed rate. The inference component includes an inference engine that can be established using AI techniques such as fuzzy logic, neural network or fuzzy-neural network techniques.

[0024] In a first embodiment, a fuzzy expert system performs input fuzzification, rules application (or inference engine) and output defuzzification. The input fuzzification translates the system-input variables into universe of input memberships. There are two common methods used for an inference engine which are Max-Product method and Max-Min inference method. The difference is the aggregation of the rules. They use multiplication and truncation of the output fuzzy set with the yielded result respectively. The defuzzification process is defined as the conversion of a fuzzy quantity, represented by a membership function, to a crisp or precise quantity. There are two common methods for defuzzification of fuzzy quantities which are the Max method and Centroid method. Two separate models have been developed to yield the desired output for wrought carbon steel turning process which are the Cutting Speed Fuzzy Model and the Feed Rate Fuzzy Model.

[0025] For the Cutting Speed Fuzzy Model, the hardness of work piece material and the depth of cut are the inputs and cutting speed is the output of the fuzzy model. FIGS. 2 and 3 show the fuzzy expressions for work piece material hardness, depth of cut and cutting speed respectively. FIG. 4 shows the range of the fuzzy input membership functions and the output membership functions. All membership functions (both inputs and outputs for all models) are of triangle shape and are well distributed. Four fuzzy models with different fuzzy rules amongst themselves are developed. The first model with fuzzy rules shown in FIG. 6 was designed for high-speed steel tools. Other models were then designed to get optimum results (cutting speed) for an uncoated indexable carbide tool, an uncoated brazed carbide tool and a coated carbide tool. The rules set for these tools are shown in FIGS. 5 to 7. The search for the optimum model for every

cutting tool has been limited to the variance in rules only, where the input and output fuzzy sets are maintained and linearly proportionate among two types.

[0026] The Feed Rate Fuzzy Model is a single-input-single-output fuzzy model. The fuzzy shape of the input and output fuzzy sets is that of a triangle, but not of an equal-sided triangle. FIG. 8 shows the fuzzy expression for both the input and output membership functions. The model applies straightforward fuzzy rules, which are

[0027] i. If the depth of cut is very shallow then the feed rate is very slow

[0028] ii. If the depth of cut is shallow, then the feed rate is slow

[0029] iii. If the depth of cut is medium, then the feed rate is medium.

[0030] iv. If the depth of cut is deep, then the feed rate is fast

[0031] v. If the depth of cut is very deep then the feed rate is very fast

[0032] Unlike the Cutting Speed Fuzzy Model, the Feed Rate Fuzzy model maintains the fuzzy rules as a common rule while changing the input and output membership functions in order to yield the best results. The high-speed steel fuzzy model is set as the reference membership function throughout the new membership function development for other tools.

[0033] The following equation E1 is used to define the input and output fuzzy sets of other tool type models.

E1

$$V_{new} = V_{new,min} + \left(\frac{V_{std} - V_{std,min}}{V_{std,max} - V_{std,min}} \right)^n \times (V_{new,max} - V_{new,min})$$

[0034] The corresponding new value of a membership function (input and output as well), V_{new} is yielded from the value of the input and output membership functions of the High-Speed Fuzzy Model, V_{std} (so named reference fuzzy model), and $V_{\text{new,max}}$, $V_{\text{new,min}}$, $V_{\text{std,max}}$ and $V_{\text{std,min}}$ are the maximum and minimum values of the new membership functions and the standard membership functions, respectively. The exponential value, n , will determine the pattern of the yielded membership function. $n=1$ will lead to a linear proportional relationship. To obtain a better correlation, the value of n is obtained from Equation E2 instead of being a constant.

$$\text{E2} \quad n = \begin{cases} A - \left(\frac{V_{\text{std}} - V_{\text{std,min}}}{(V_{\text{std,max}} - V_{\text{std,min}}) \times R} \right)^{P_1} \times (A - B), & -V_{\text{std}} \leq V_{\text{std,min}} + (V_{\text{std,max}} - V_{\text{std,min}}) \times R \\ 1 - \frac{V_{\text{std,min}} + (V_{\text{std,max}} - V_{\text{std,min}}) - V_{\text{std}}}{(V_{\text{std,max}} - V_{\text{std,min}}) \times (1 - R)^{P_2} \times (A - B)}, & -V_{\text{std}} > V_{\text{std,min}} + (V_{\text{std,max}} - V_{\text{std,min}}) \times R \end{cases}$$

[0035] The value n will be equal to A at the lowest V_{std} valued and gradually decrease to B (which must be a value smaller than unity) when V_{std} is at the required ratio of the range, R . The changing pattern of n is dependant on the constant P_1 . Then n will gradually increase to unity at the maximum value of V_{std} . Again, the changing pattern of n is depending on constant P_2 .

[0036] In the second embodiment the genetic optimization is used to further optimize the fuzzy rules. Major operations of genetic algorithm depend on random choice. A random number generator class is developed. It consists of three main functions.

[0037] They are: 1) to generate a random real number from 0 to 1; 2) to generate a random number from a user given start to a user given end integer value; and 3) to give a green or red light for a user given probability.

[0038] Genetic optimization of fuzzy rules has been carried out with the help of an object oriented genetic optimization library (GOL). It consists of several useful and interconnected classes. To evaluate the fuzzy result, the Fuzzy Set Handling (FSH) class has to be included. FIG. 10 shows a general system flow of the genetic optimization process for fuzzy rules design.

[0039] The population needs to be initialized. Information such as the number of alleles and the length of each allele are predetermined. The GOL use bit-wise interpretation, which means, the length of a particular allele is expressed in term of bits. If an allele carries a possible value from 0 to 7 (or 8 possible features), the length of allele is 3. The fuzzy models consist of 5 fuzzy sets for all inputs and 15 fuzzy output sets. The system required 25 fuzzy rules with 15 possibilities each. For initialization, 25 alleles are required and the length of each allele will be 4 bits to cope with 15 possible values. Besides initialization of the alleles, the probabilities of crossover and mutation, and the size of population are needed. Proper size of the population and probabilities will yield better optimization results in terms of speed.

[0040] The population consists of 80 sets of fuzzy rules. The crossover and mutation probabilities are set as 0.6 and 0.009, respectively. The initial members of the population are required. The fuzzy rules from Fuzzy Logic are assigned as one of the initial members. The rest of the rules are generated automatically and randomly. Initialization of FSH class in FIG. 1, referring to the construction of the FSH class, includes building the membership functions of inputs and outputs. With the FSH class, the fitness of each member of the initial population has to be calculated. The one with better fitness stands a better chance of success in reproduction. In the process, crossover and mutations may occur and thus, create new species. Crossover operation is single-point operation. The mutation operation is a random change in an allele,

instead of a bit in the allele. The new generation has a tendency towards better quality. The reproduction continues until the population is completed. The whole process repeats until the maximum number of generation is obtained. Fitness is obtained through Equation E3.

E3

$$Fitness = K - \sum_{i=0}^n \frac{(abs_error\%)_i}{n+1} - \sum error_factor$$

[0041] The value of i in E3 is the numbering of the predetermined inputs starts from 0. Thus, value n is the ending number of the predetermined inputs, in this case 79. Error factor is for rules violation penalty. The value K is an arbitrarily selected positive value. It must be foreseen to be more than a then-achievable maximum summation of the absolute error percentage mean and the error factors. Generally, with lower mean absolute error percentage, the fitness value is higher and nearer to K . For the final consideration of the fitness in the competition of reproduction selection, Equation E4 is employed.

E4

$$FFitness_i = Fitness_i - \min(Fitness_0 \dots Fitness_n)$$

[0042] The Genetic Optimization Class allows control of fuzzy rules patterns, which means that the designer can provide his/her expertise in specifying the relationship between the inputs and the outputs. Each violation will cause a penalty value, thus more violations will cause lower fitness. The user can set the penalty value. The penalty value must not be too small, since a small penalty value will not cause a significant effect. It must not be too large either, since too large a penalty value will make the whole optimization process inefficient. The summation of all penalty values must not be greater than the difference between value of K and the mean of all absolute error percentages.

[0043] In the third embodiment, a non-linear artificial neural network, which consists of both the

novel product neurons developed by the inventor and the common summation neurons, is designed to prescribe machinability data for turning process of wrought carbon steel with different tools. Two feed-forward neural networks are suggested. FIG. 11 shows an artificial neural network suggested. The network incorporates simple non-linearity of second input (the depth of cut). Two different neurons are indicated with different symbols, with " Σ " representing a normal summation neuron and " Π " representing the new Product Neuron. The transfer function of each respective neuron is placed following the neuron symbol. "PL" stands for a pure linear transfer function and "P2" stands for a squared transfer function. The design of the network depends on the relationships among the parameters. The artificial neural networks suggested are supervised feed forward networks. No closed synaptic loop is used.

[0044] The summation neuron is the classical neuron, which is used in most of the present neural systems. A neuron with R inputs and bias, b has the network summation value, n computed by the following equation:

$$n = Wp + b$$

[0045] where W and p are the weight matrix and input matrix, which store corresponding weights and input values respectively. The transfer function, f may be a linear or non-linear function of n . The output neuron, a, can be written as below:

$$\begin{aligned} a &= f(n) \\ &= f(W_{p+b}) \end{aligned}$$

[0046] The product neuron is introduced by the inventors to deal with non-linearity and establish cross-neuron relations in a simpler way. The neuron may be hooked with multiple neurons but only weighted once. Similar to a summation neuron, it holds a bias as an independent variable.

The n is calculated with Equation E5.

$$\text{E5} \quad n = \left(\prod_{i=0}^R P_i \right) w + b$$

[0047] To avoid complication in design of the learning methodology and data storage, the neuron is deliberately planned to have one overall weight, instead of an individual weight for each input. Thus, processing (including learning) of a single Product Neuron is simpler than processing of a summation neuron.

[0048] The final output(s) of the network are computed with propagation forward through the network from the input(s). The forward propagation process can be expressed with the Equations 6, 7 and 8, where a^m is the output matrix of all neurons in layer m . The initial matrix a^0 represents the input values. The outputs of the whole network, which has M layers, are gathered as matrix a . The matrix f^m collects transfer functions of all neurons in layer m .

$$\text{E6} \quad a^0 = p$$

$$\text{E7} \quad a^m = f^m (W^m a^{m-1} + b^m) \text{ for } m = 1, 2, \dots, M$$

$$\text{E8} \quad a = a^M$$

[0049] The back propagation algorithm is a generalization of the least mean squared method to provide learning ability to the network. It is capable of handling multi layer network. The classical back propagation algorithm is summarized in this section.

[0050] The first step of the algorithm is to find the final output of the network with forward propagation. Then the sensitivity of each neuron is calculated with backward propagation through the network from the outermost layer, which yields the output(s). Recurrence relation is established among the neurons and chain rules are used to obtain the sensitivities. The neuron sensitivities of last layer are calculated in Equation E9 in matrix form.

$$s^M = -2F^M(n^M)(t-1)$$

[0051] Matrix F^m consists of all target outputs. The $F^m(n^m)$ is the sensitivity function of respective neuron and is expressed as the Equation 10, where S_m is the number of neurons in layer m . For other layers, the Equation 11 is used.

$$E10 \quad F^m(n^m) = \begin{bmatrix} \frac{\partial f^m(n_1^m)}{\partial n_1^m} & 0 & \dots & 0 \\ 0 & \frac{\partial f^m(n_2^m)}{\partial n_2^m} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{\partial f^m(n_{s^m}^m)}{\partial n_{s^m}^m} \end{bmatrix}$$

$$E11 \quad s^m = F^m(n^m)(W^{m+1})^T s^{m+1}$$

[0052] Finally the weights and biases are updated with selected update method, which may be

referred as learning algorithm.

[0053] Weights and biases are updated with Approximate Steepest Descent method in this paper. The weights and biases are updated as shown in Equations 12 and 13.

$$\text{E12} \quad W^m(k+1) = W^m(k) - \alpha s^m (a^{m-1})'$$

$$\text{E13} \quad b^m(k+1) = b^m(k) - \alpha s^m$$

[0054] Momentum is used to smooth out the oscillations in the trajectory towards the optimum location. Equations 14 and 15 are used to achieve such task, where $\Delta W^m(k)$ and $\Delta b^m(k)$ are the k th change of weights and biases at layer m . The learning rate and the momentum coefficient are denoted as α and γ respectively.

$$\text{E14} \quad \Delta W^m(k) = \gamma \Delta W^m(k-1) - (1-\gamma) \alpha s^m (a^{m-1})^T$$

$$\text{E15} \quad \Delta b^m(k) = \gamma \Delta b^m(k-1) - (1-\gamma) \alpha s^m$$

[0055] From the mathematical point of view, the BP learning with SD method only promises a general minima achievement, whether a global, local or weak minima. It is guaranteed to converge to a solution that minimizes the mean squared error, as long as the learning rate is not too large. Typically, the initial weights and biases are chosen to be small random values. In this way, one can stay away from a possible saddle point at the origin. It is also useful to try several different guesses in order to be sure that the algorithm converges to a global minimum point.

[0056] A new retraining scheme is suggested to move the solution away from any local or weak minima during the training of an artificial neural network. If the network is found having weak response towards a particular training set, extra retrainings are enforced until the network's performance index of the training set is under a predefined-level, α . Generally, implementation of the scheme incurs a certain level of interruption towards the network (refer to the weight and bias values). Thus, one may still observe a sudden fluctuation on the performance surface even if the solution reached its global minima. This can be solved by incorporating a simple global-minima tracking procedure. Implementation of the RR does not require Variable Learning Rate (VLR) algorithm. The involvement of the VLR will slow the overall efficiency of the RR scheme, as the main purpose of the RR scheme is to bring the solution out of the local or weak minima towards the global minima. Initial studies by the inventors show that the VLR with RR cannot bring the solution out of the local or weak minima efficiently. The RR scheme requires a stable learning rate to obtain overall effect during the training process. The SD method alone is good enough to carry the task. The squared error limit, α , is calculated with Equation 16, where σ_{initial} and σ_{end} are the user defined values which indicate the squared error limit at the initial and the end of the training respectively. $i\text{Train}$ and $i\text{Train}_{\text{max}}$ are the current number of trainings and the maximum number of trainings desired respectively. As general practice, the value of σ_{initial} should be larger than σ_{end} .

E16

$$\sigma = \frac{\sigma_{\text{initial}} - (\sigma_{\text{end}} - \sigma_{\text{initial}}) \times i\text{Train}}{i\text{Train}_{\text{max}}}$$

[0057] The scope of the invention should be defined only in accordance with the claims that follow. In the following claims, reference characters used to designate claim steps are provided for convenience of description only, and are not intended to imply any particular order for performing the steps.